IN THE CLAIMS

Please amend the following claims:

1. (Amended) A method for aligning an [macro] instruction stream comprising:

rotating data bytes of the [macro] instruction stream; and

shifting the data bytes to a [the] start of the [a macro] instruction based upon a length of an immediately prior [macro] instruction.

3. (Amended) The method of claim 1 further comprising:

receiving a length of an immediately prior [macro] instruction from a length decode logic unit.

4. (Amended) The method of claim 1 further comprising:

storing [macro] instruction stream cache lines in alignment buffers prior to rotating the [macro] instruction stream.

5. (Amended) The method of claim 1 wherein said shifting shifts to an exact start of the [macro] instruction.

7. (Amended) Logic for aligning an [macro] instruction stream comprising:

a rotator logic unit for rotating data bytes of the [macro] instruction stream;

a shifter logic unit for shifting the data bytes to a [the] start of the [a macro] instruction based upon a length of an immediately prior [macro] instruction.

9. (Amended) The logic of claim 7 further comprising:

a length vector for providing the length of an immediately prior [macro] instruction.

10. (Amended) The logic of claim 7 further comprising:

alignment buffers for storing [macro] instruction stream cache lines for use by the rotator logic unit.

11. (Amended) A processor to align an [macro] instruction stream comprising:

a rotator logic unit for rotating data bytes of the [macro] instruction stream;

a shifter logic unit for shifting the data bytes to a [the] start of the [a macro] instruction based upon a length of an immediately prior [macro] instruction.

12. (Amended) The processor of claim 11 further comprising:

a length vector for providing the length of an immediately prior [macro] instruction.

13. (Amended) A system for aligning an [macro] instruction stream comprising:

means for rotating data bytes of the [macro] instruction stream; and

means for shifting the data bytes to a [the] start of the [a macro] instruction based upon a length of an immediately prior [macro] instruction.

Please add the following claims:

14. The method of claim 1 further comprising:

determining a length of a current instruction.

15. The method of claim 14 wherein the length of the current instruction is based upon a length of an opcode and a length of immediate data.

16. The method of claim 14 further comprising:

determining if an opcode extension byte is required to determine the length of the current instruction.

17. The method of claim 6 further comprising:

determining a memory address displacement length.

18. The method of claim 17 further comprising:

determining an anticipatory length of the memory displacement for a one-byte opcode; and

determining an anticipatory length of the memory displacement for a two-byte opcode.

19. The method of claim 18 further comprising multiplexing the anticipatory length for the one-byte opcode and the anticipatory length for the two-byte opcode to determine the length of the memory displacement.